AN ASYNCHRONOUS NETWORK-ON-CHIP ROUTER WITH LOW STANDBY POWER

A Thesis

by

AMR N. ELSHENNAWY

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Sunil P. Khatri
Committee Members,	Jiang Hu
	Anxiao Jiang
Head of Department,	Chanan Singh

September 2014

Major Subject: Computer Engineering

ABSTRACT

The Network-on-Chip (NoC) paradigm is now widely used to interconnect the processing elements (PEs) in a chip multiprocessor (CMP). It has been reported that the NoC consumes about a third of the total power consumption of the multi-core processor. To address this, asynchronous NoC routers have been proposed, to eliminate the clocking power associated with the NoC implementation, which is typically a large fraction of the NoC power consumption. In this work, we present a technique to reduce the standby power of a state-of-the-art asynchronous NoC router. In our approach, the router is put in a known input state when idle, and each gate in the unmodified router is replaced by a logically equivalent gate whose supply pin is connected to a PMOS device with a high threshold voltage in case its output in the idle state was 0. On the other hand, if the output of the unmodified gate in the idle state was 1, it is replaced by a logically equivalent gate whose ground terminal is connected to a NMOS device with a high threshold voltage. Our router is inserted in an NoC, and verified logically for correct routing functionality. We also simulated it at the circuit level using a 45nm fabrication technology, and show that it has a low wake-up time from sleep, and a minimal steady-state routing delay (13%) and area (23%) overhead, and a $8.1\times$ lower standby power, when compared to an unmodified asynchronous NoC router, which was also implemented. Our leakage improvement is achieved in part by using a novel method to control the leakage of the inverter chain used to drive the sleep signal, something which that is not possible with traditional leakage reduction techniques.

To my family

TABLE OF CONTENTS

LIST OF TABLES

TABLE Page

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

The number of processing elements (PEs) on modern chip multiprocessors (CMPs) continues to grow with the scaling of VLSI fabrication technology. As a consequence, it becomes increasingly important to connect the PEs of the CMP in a scaleable and efficient manner. The shared bus based communication paradigms scale poorly, and exhibit high arbitration complexity and low bandwidth. Also, adding more PEs to a shared bus adds parasitic capacitance, degrading speed further.

I-A. Network-on-Chip (NoC)

The Network-on-Chip (NoC) paradigm has emerged as the practical alternative to the traditional shared bus, since it is able to scaleably improve the bandwidth of CMP communication. Each link in the NoC can be used simultaneously, allowing a high level of parallelism. The NoC also allows data to be pipelined, providing a much greater aggregate bandwidth than shared buses. A typical NoC is modular in its design, with the basic components such as network interfaces, routers and links. These units are designed once and then replicated across the chip, reducing design complexity.

A large fraction of the total CMP power is consumed by the NoC. For example, the NoC in the RAW [1] system consumes as much as 36% of the total power of the CMP. Hence, it is critical to reduce the power of the NoC.

Recent studies [2] show that asynchronous NoCs are an appealing solution to tackle the synchronization challenge in modern multicore CMP systems, using the Globally Asynchronous Locally Synchronous [3] (GALS) design paradigm. Due to the difficulty in distributing a global clock as technology scales and processing, temperature and voltage (PVT) variations dominate, the GALS paradigm has been gaining favor among researchers

and practitioners. In the GALS paradigm, individual computing units (PEs of the CMP) are synchronous, while communication among PEs is performed asynchronously. This naturally suggests that the routers in a CMP be designed in an asynchronous manner, while the PEs are synchronous. This allows the designer to absorb the heterogeneity of timing constraints in the system interconnect of such systems.

I-B. Asynchronous Network-on-Chip

Asynchronous NoCs, then, are the best means to accomplish such a GALS inspired CMP. In particular, in an asynchronous NoC, the router in the NoC is implemented in an asynchronous manner. They have several attendant benefits, such as a) they provide average-case instead of worst-case, guard-banded performance, b) they save on power since they do not need a clock signal, and hence the switching power of the clock tree is eliminated, and c) robustness to PVT variations due to the asynchronous nature of the design.

There are some downsides to asynchronous design (and hence asynchronous NoCs). The main issue is that they are harder to design and test in general than the synchronous NoCs. There is scant tool support when it comes to design, verification and testing of asynchronous circuits. As a consequence, designers rely on a full- or semi-custom approach for the design of such circuits.

I-C. Leakage Power

Leakage power has become a significant issue in modern VLSI design. As process feature sizes and operating voltages of VLSI designs shrink, (sub-threshold) leakage currents in modern VLSI designs become dominant. When a circuit such as an NoC router is in the standby mode of operation, its power consumption is due to the sum of leakage currents in all its devices.

The leakage current for a PMOS (NMOS) transistor is the I_{ds} of the device when it is in the *cut-off* or *sub-threshold* region of operation. This current [4] is governed by the equation:

$$I_{ds} = \frac{W}{L} I_0 e^{(\frac{V_{gs}-V_T-V_{off}}{nv_t})}(1 - e^{(-\frac{V_{ds}}{v_t})}) \tag{1.1}$$

Here I_0 and V_{off}[1] are constants, while $v_t = \frac{kT}{q}$ is the thermal voltage (26mV at 300°K) and n is the sub-threshold swing parameter. Also, T is the absolute temperature, k is the Boltzmann constant and q is the electron charge.

From the above, leakage currents increase exponentially with decreasing threshold voltages (which are typically maintained at a fixed fraction of supply voltage). Hence, as supply voltages are reduced, leakage increases exponentially. They also increase exponentially with temperature, which is another significant problem. This is a major concern for VLSI design in the nanometer realm [5]. It is expected that leakage power is comparable to switching power in recent VLSI technologies [5, 6].

A closer look at Equation 1.1 shows that I_{ds} is significantly larger when $V_{ds} \gg v_t$. This is because of two effects: a) the last term of equation 1.1 is close to unity and b) with a large value of V_{ds}, V_T would be lowered due to drain induced barrier lowering[2] (DIBL) [7, 4]. Therefore, leakage can be reduced by ensuring that the supply voltage is not applied across a single device.

I-D. This Work

To reduce leakage power consumption, the authors of [8] devised a circuit design approach in which a circuit block is driven with a fixed input vector in the standby mode of operation.

[1] Typically $V_{off} = -0.08V$

[2] V_T decreases approximately linearly with increasing V_{ds}

Based on the static outputs of each gate for this input vector, the gates were modified such that if a gate had a "1" output during standby, its ground terminal was connected to a high-V_T NMOS switch (the modified cell is referred to as an *H cell*). Conversely, if a gate had a "0" output during standby, its supply terminal was connected to a high-V_T PMOS switch (the modified cell is referred to as an *L cell*). This ensured that the supply voltage, during standby, was applied across at least two devices, reducing leakage significantly, while ensuring that the logic values of each gate output did not float, allowing for fast recovery (wake-up) from the standby state once normal operation has to be resumed.

In this thesis, we apply the leakage reduction ideas of [8] to the asynchronous router design that was proposed in [2]. The asynchronous router of [2] demonstrated a significant reduction in area, power and energy-per-flit over *xpipesLite* [9], an existing state-of-the-art synchronous router. The approach of [2] utilizes a two-phase bundled data protocol on the inter-router links, as well as within the router itself. By allowing this router to sleep between packets using the design approach of [8], we are able to demonstrate a 8.1$\times$ reduction in leakage power, with a minimal wake-up time, and router delay once the router has been woken up from its standby state.

The key contributions of this thesis are:

- We modify the asynchronous NoC router of [2] to make it compatible with the leakage reduction methodology of [8].
- We verify the correct operation of the modified NoC at the logic level, by validating its correctness in a 4$\times$4 NoC.
- We simulated the design of the modified and unmodified [2] NoC in HSPICE using a 45nm predictive technology [10], and show that the modified design exhibits 8.1$\times$ lower leakage, with a low wake-up time of $\sim$870ps, and a steady-state flit routing speed penalty of 13% over the unmodified design of [2].

- We use a novel technique to reduce the leakage of the inverter chain that drives the sleep signal to the router blocks, which cannot be realized by existing leakage control approaches.

- Unlike traditional leakage control approaches, our approach allows the Input Port Modules (IPMs) and the Output Port Modules (OPMs) to sleep independently. With traditional leakage control approaches, then entire router must be put in sleep mode at once, reducing the leakage power reduction that can be availed in practice.

The rest of the thesis is organized as follows. In Chapter II, we discuss previous work in the area of asynchronous NoC design, and leakage reduction techniques. Chapter III describes our approach, and Chapter IV presents experimental results. Chapter V presents our conclusions.

CHAPTER II

PREVIOUS WORK

In this chapter, we first discuss some of the key works in asynchronous design, especially in the context of NoCs. We next discuss other NoC topologies, and then cover previous work in the area of low leakage VLSI design.

For NoC designs, the GALS paradigm [3] has become popular in recent times. Some of the more efficient GALS based NoC approaches have been published are [11, 12, 13]. Our work is based on the recent efficient asynchronous NoC reported in [2], which uses a very efficient two-phase bundled data protocol, based on transition signaling. Most of the previous approaches, in contrast, utilize four-phase return-to-zero protocols, which require two round-trip messages to be sent per transaction, making them less practical for a state-of-the-art NoC design. The Mousetrap pipeline protocol [14] that is used in [2] is used in our work as well.

II-A. Network-on-Chip (NoC) Topologies

The simplest and most common NoC topology is the 2D mesh [15]. The mesh topology consists of a 2D mesh of wires, with routers at the intersections of horizontal and vertical wires. Each router consists of five ports, with one port connected to the PE and the others connected to the closest neighboring routers in the 4 cardinal directions. Our work is based on a router for a 2D mesh topology. Other topologies that have been published include the torus architecture [16], the octagon architecture [17], a 2D flattened butterfly [18], the Fat Tree [19], and the ring [20]. Due to the ubiquity of the 2D mesh, and the existence of an efficient realization of an asynchronous 2D mesh router in [2], our work focuses on the mesh topology.

II-B. Leakage Reduction Techniques

Several means for reducing leakage power have been proposed in recent times. In [21], the authors propose a dynamic threshold MOSFET design for low leakage applications. The drawback of this approach is that it is only applicable in situations where VDD is much lower than the diode turn-on voltage. This technique is typically useful for partially depleted silicon-on-insulator (SOI) designs. Another leakage reduction technique is the Variable-threshold [22, 23] (often called VTCMOS) approach. In this scheme, the device threshold voltages are controlled by adjusting the device bulk voltage dynamically, requiring complex analog circuitry.

More traditional design approaches have suggested the use of dual threshold devices [24] in an MTCMOS (multi-threshold CMOS) configuration[1]. Cell inputs and outputs as well as bulk nodes float in an MTCMOS design operating in standby mode. As a result, an MTCMOS designs can have a large range of leakage currents, and more problematically, their wake-up times from the standby (equivalently referred to as sleep) state is high. Variants of the MTCMOS approach are the *Header-only* and *Footer-only* approaches. In the *Header-only* approach, a high-threshold PMOS header is used in series with the supply terminal of a design. In the *Footer-only* approach, a high-threshold NMOS header device is used in series with the ground terminal of a design. These techniques are slightly more easy to implement than MTCMOS, but retain the problems associated with MTCMOS (which were mentioned above). A scheme to fix these problems, while still yielding the leakage improvement of MTCMOS is the HL approach [8]. In this approach, a specific input vector (*sleep vector*) is applied to the circuit when it is in standby operation. The circuit is simulated for this input vector. Based on the static outputs of each gate for the sleep vector, the supply and ground terminals of each gate are modified. If a gate G had a "1" output

[1] MTCMOS utilizes both high-threshold NMOS and PMOS power supply gating devices

during standby, then it's NMOS stack leaks, and hence its ground terminal is connected to a high-V_T NMOS switch. The modified cell is referred to as an *H cell*). On the other hand, if a gate had a "0" output during standby, then it's PMOS stack is leaking during standby. Hence, its supply terminal is connected to a high-V_T PMOS switch to reduce leakage. The modified cell is referred to as an *L cell*). The HL approach has improved delay characteristics compared to MTCMOS (since the delay of only one output transition of a gate is degraded). Also, the gate outputs do not float during standby, reducing wake-up delay as well as wake-up power consumption significantly. As a result of these benefits, we utilize the HL approach to reduce the leakage of the asynchronous router.

In [25, 26], the authors address the problem of finding the best vector to utilize when the circuit is in standby mode. It was shown that the best vector has a leakage of about 50-75% of the worst leakage vector. Our approach uses a random vector for standby operation. Our leakage improvements results could potentially be improved by 2× if a leakage vector selection algorithm was employed.

II-C. Leakage Reduction for Asynchronous Circuits

In [27], the authors applied an HL-like technique to reduce power in an m-out-of-n design, using a 4-phase asynchronous handshaking protocol. They use a 90nm process technology, obtaining a 94% reduction in leakage power, with a 30% delay penalty, averaged over 5 small ISCAS89 benchmark circuits. In contrast, we demonstrate our leakage reduction technique for a significantly larger design (an asynchronous router), implemented using a superior 2-phase bundled data protocol implemented in a 45nm technology. Our results are significantly improved (8.1× leakage reduction compared to the regular router, with a modest 13% steady-state flit routing delay penalty). In another work [28], the authors applied leakage reduction techniques for asynchronous circuits, for an Advanced Encryption

Standard (AES) design implemented in a 90nm process technology. Unlike our approach, they obtained a minimal leakage reduction of about 30%, with a 20% delay penalty.

CHAPTER III

OUR APPROACH

In this chapter, we first introduce asynchronous handshaking protocols. We next discuss the traditional leakage reduction techniques, followed by a discussion on the HL approach used in this work. Next, we present a brief overview of the asynchronous NoC router [2] we based our work on. Finally we discuss the Input Port Module (IPM) and Output Port Module (OPM) modules of the asynchronous router, along with a discussion on how we modified them to reduce their leakage currents.

III-A. Asynchronous Handshaking Protocols

In asynchronous communication, the sender uses its REQ signal to indicate the data it wishes to send to the receiver is ready. The receiver uses its ACK signal to acknowledge to the sender that the data has been successfully received. In general, there are two different implementation of this handshake: the 4-phase protocol and the 2-phase protocol.

III-A.1. Four-Phase Signalling Protocol

The 4-phase protocol (also called the return-to-zero or RZ protocol) is shown in Figure III.1. The sender and receiver signals are nominally low. The sender signals that it has data to send by raising its REQ signal. When the receiver has received the data, it raises its ACK signal. This results in the sender dropping its REQ signal, which in turn causes the receiver to drop its ACK signal in the RZ phase. Once both the REQ and ACK are low again, a new transaction may begin. The key disadvantage of the 4-phase protocol is the superfluous return-to-zero transitions that cost unnecessary time and energy.

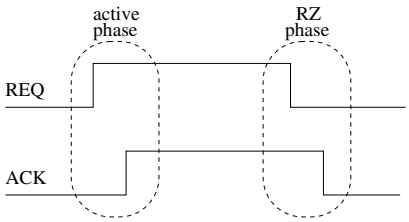


Fig. III.1. 4-Phase Asynchronous Protocol (RZ)

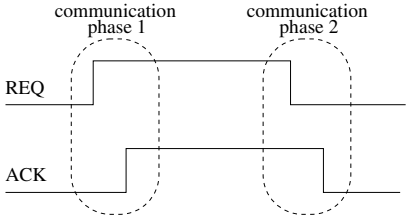


Fig. III.2. 2-Phase Protocol (NRZ)

III-A.2. Two-Phase Signalling Protocol

The 2-phase asynchronous handshake protocol shown in Figure III.2 avoids the additional transitions of the 4-phase protocol. Assume that the sender's REQ and the receiver's ACK are nominally low (high). The sender signals that it has data to transmit by transitioning its REQ to high (low). Once the receiver has received the data, it transitions its ACK to high (low). Note that the two additional "return-to-zero" phases are absent, and are instead used to implement a separate transaction. As a consequence, the 2-phase protocol generally provides higher performance and lower energy consumption than the 4-phase protocol.

III-B. Traditional Leakage Reduction Techniques

With decreasing process feature sizes, a significant increase in leakage power as a fraction of total power has led to an increased focus on leakage current control [24, 21, 23, 22, 25, 26, 8]. Each of these approaches increase V_T values to reduce leakage, in a static (at the

time of fabrication) or dynamic (by increasing V_T via bulk voltage control). In this work, we utilize the approach of [8], due to its simplicity, leakage improvements and fast wake-up time.

The simplest approach, multi-threshold CMOS (MTCMOS), connects the ground (and supply) terminal of a cell to a high-threshold footer NMOS (and high-thresold header PMOS) transistor. Note that since the PMOS device threshold voltages are negative, the threshold voltage of the PMOS header is actually lower than that of other PMOS transistors in the circuit. In the remainder of this writeup, we refer to the absolute value of threshold voltages of any device. Consider the unmodified circuit shown on the left of Figure III.3. The MTCMOS equivalent of this circuit is shown on the right of Figure III.3. Note that the supply (and ground) signals are driven to each cell through the header (footer) devices. In practice, a single header or footer device can be used for a large number of logic cells.

Variants of MTCMOS that are commonly used in industrial practice are the *Header-only approach* (see Figure III.4) and the *Footer-only approach* (see Figure III.5). They are generally preferred since they involve the use of only high-threshold header (footer) devices, unlike MTCMOS, which requires both.

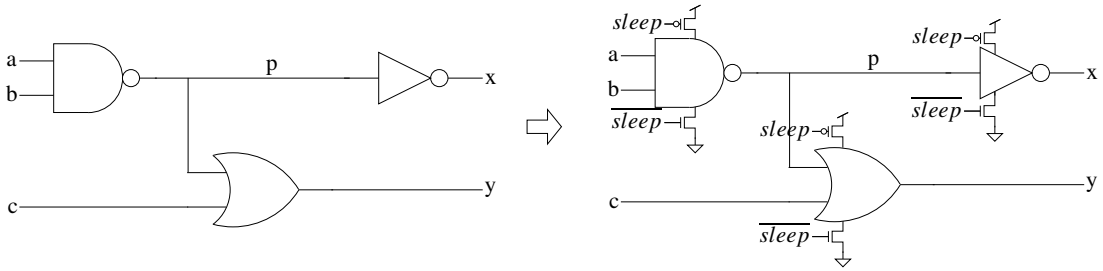


Fig. III.3. Low-leakage MTCMOS Circuit

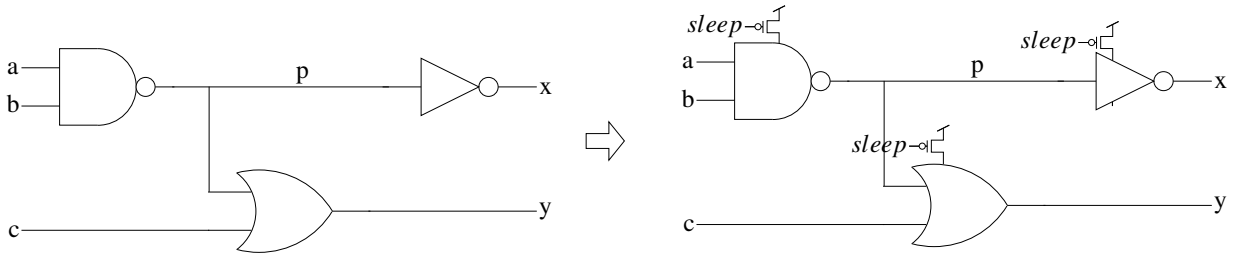


Fig. III.4. Low-leakage Header-only Circuit

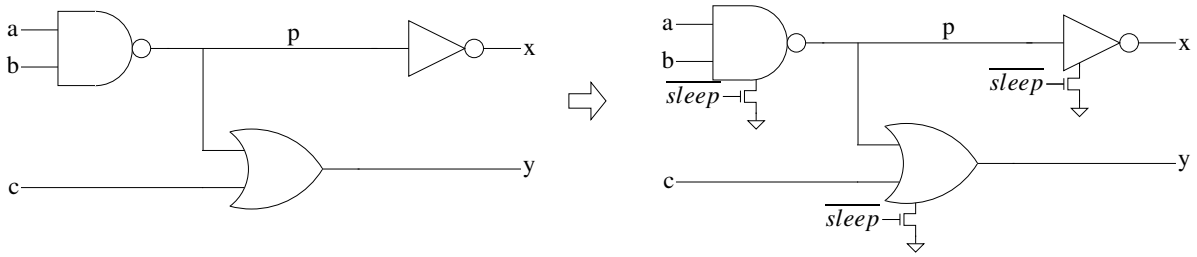


Fig. III.5. Low-leakage Footer-only Circuit

III-C. Leakage Reduction with the HL Approach

The disadvantage of the MTCMOS, Header-only and Footer-only approaches is that gate outputs float to non-rail values during the *standby* (equivalently referred to as *sleep*) mode of operation. This results in a variability in the leakage of the circuit, and large delays and inrush currents during wake-up. To fix this, the HL approach [8] ensures that the gate outputs maintain rail values during sleep mode. The circuit primary inputs are driven to a fixed vector during sleep. Simulating the circuit under this vector yields the gate outputs for each gate in the design (see the left part of Figure III.6). The low-leakage HL circuit is obtained by:

- If a gate G has an output value of "1" when the sleep vector is applied to the circuit, then the ground pin of G is connected to the high-threshold NMOS footer device.

The modified gate is referred to as a *H cell*, and its leakage is reduced during standby since there are at least two devices between its output and the ground terminal, one of which has a high threshold voltage.

- If a gate *G* has an output value of ”0” when the sleep vector is applied to the circuit, then the supply pin of *G* is connected to the high-threshold PMOS header device. The modified gate is referred to as a *L cell*, and its leakage is reduced during standby since there are at least two devices between its output and the supply terminal, one of which has a high threshold voltage.

The HL conversion of the circuit on the left part of Figure III.6 is shown on the right side of the same figure. Note that the header and footer devices can be shared across several gates in practice. Our low-leakage asynchronous router shares header and footer devices across several gates.

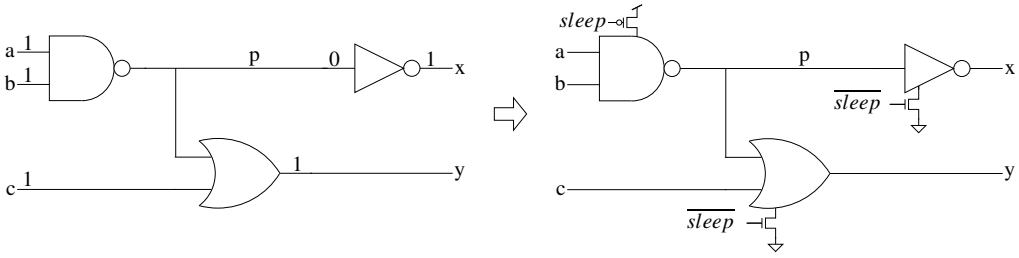
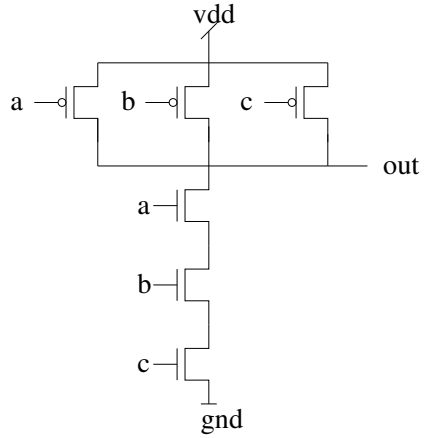
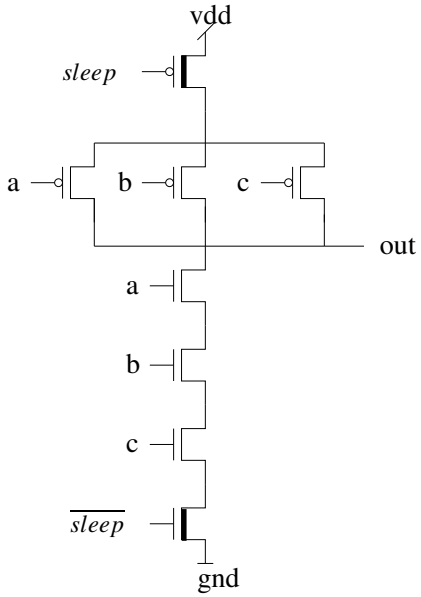


Fig. III.6. Low-leakage HL Circuit

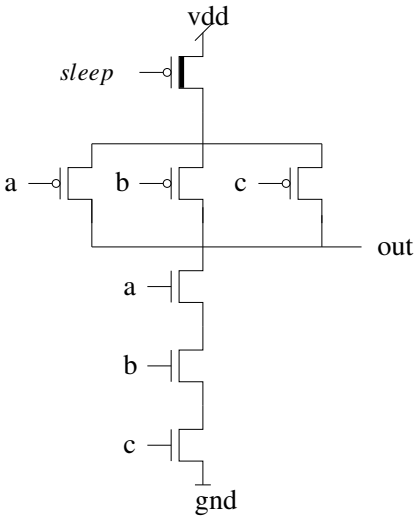
Figure III.7 shows the schematic of a NAND3 gate, along with the corresponding low-leakage MTCMOS version of the same gate, and the H and L versions as well.



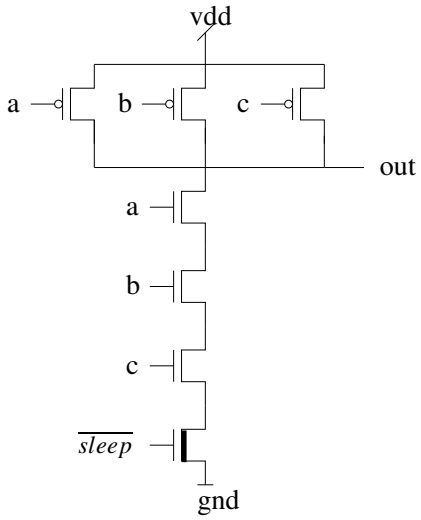
Regular 3−input NAND



MTCMOS implementation of a 3−input NAND



L variant of a 3−input NAND



H variant of a 3−input NAND

Fig. III.7. NAND3 Gate - Unmodified, MTCMOS, H Variant and L Variant

III-D. Overview of Asynchronous Router

The asynchronous router we base our work on is a re-implementation of the asynchronous router reported in [2]. This state-of-the-art router utilizes a two-phase bundled data routing protocol. In the NoC, each PE has an associated asynchronous router. The router has 5 input ports and 5 output ports. The architecture of the router is shown in Figure III.8. Routing is performed using 5 Input Port Modules (IPMs) and 5 Output Port Modules (OPMs). One input port and one output port are used to communicate with the associated PE, while the other 4 input and output ports connect to the corresponding ports of the asynchronous routers in the North, South, East and West directions respectively. IPM0 and OPM0 connect to the PE. The routers support a NoC of size up to 16×16 PEs, with an 8-bit address. A *packet* consists of a *head flit*, one or more *body flits*, and a *tail flit*. The router supports wormhole routing. In other words, once a head flit traverses a router, the routing path is reserved until the tail flit of the same packet leaves the router. Dimension-order routing (DOR-XY) is supported, with the X dimension routing being performed before the Y dimension routing. Each flit is 144 bits wide for our re-implementation of [2] as well as our low-leakage asynchronous router design.

Among the 144 bits in each flit, the 2 least significant bits indicate the flit type (11 for head flit, 10 for tail flit and 00 or 01 for body flit). The head flit reserves an additional 8 bits to store the packet destination address.

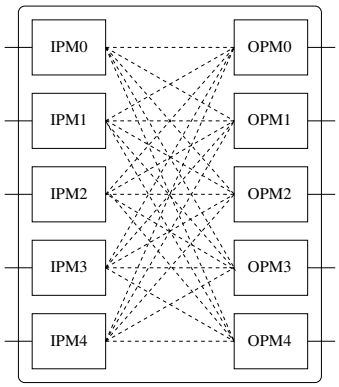


Fig. III.8. Asynchronous Router Block-level Overview

A key requirement for leakage reduction in the HL methodology is that the inputs to a design stay at a fixed, predetermined value during standby. In our implementation of the HL-based low-leakage asynchronous router, we allow the router to enter standby operation *between* packets, but not within a packet (i.e. between flits). This is because sleeping between flits would not give the power supplies enough time to reach their steady state values if we sleep between flits, since the capacitances on the supply and ground nodes in the design is high.

Also, we assume that every packet has an even number of flits. This ensures that the REQ and ACK signals have a fixed and identical value (assumed to be ”0” in our simulations) before and after a packet traverses the router. As a result, both REQ and ACK have a ”0” value during standby, allowing us to maximize leakage power improvements. Without this restriction, the REQ and ACK signals could both be ”0” or ”1” during standby, making it impossible to assign an H or L type to the gates in the fanout of the REQ and ACK signals, thereby increasing leakage power.

III-D.1. Modified Input Port Module (IPM) for Reduced Leakage

The role of the IPM is to forward flits to the appropriate OPM in the router, and handle wormhole routing. The schematic of IPM4 is shown in Figure III.9. Other IPMs are identical, with slightly altered signal indices.

The original unmodified logic of the IPM is shown in the dotted region of Figure III.9. The modifications to the IPM to enable leakage reductions are shown in Figure III.9 in the solid boxes.

Routing begins with a header flit of a packet arriving at IPM4. For the following discussion, we assume it's destination address requires that it exit from OPM1. Initially, *REQIN* and *ACKIN* are "0" as discussed. *DATAIN* is driven to IPM4, with the header flit contents. Based on the packet destination address in the header flit, the *Packet Route Selector* block determines which OPM the packet is intended for. Exactly one of the lower inputs to the 4 AND gates in the *Packet Route Selector* go high, awaiting the *REQIN* signal to go high. After a predetermined delay, *REQIN* goes high, and it is latched as REQX. The rising of REQX causes the *RouteSelected1* signal to go high, which is latched through the SR latch, so that *PacketPathEnabled4_1* goes high. This signal reserves the route from IPM4 to OPM1 until the tail flit is detected by OPM1 (indicated by the *TailPassed1_4* signal. This is how wormhole routing is implemented. Only one *RouteSelectedi* signal can be high at a given time. Note that the REQX signal is driven to all OPMs as *REQ4_i*, since *ACKi_4* are initially all "0". Once the flit is driven out from OPM1, the *ACK1_4* signal will be driven high, ensuring that *REQ4_i* gets driven to "0" for $i = 0, 2, 3$. Finally, the *ACKX* signal, which is the XOR of the four *ACKi_4* signals, goes high, indicating to the sender that a new flit can be driven on *DATAIN*. Once the tail flit of the packet passes through OPM1, the *TailPassed1_4* signal is driven high, tearing down the wormhole route and driving *PacketPathEnabled4_1* low.

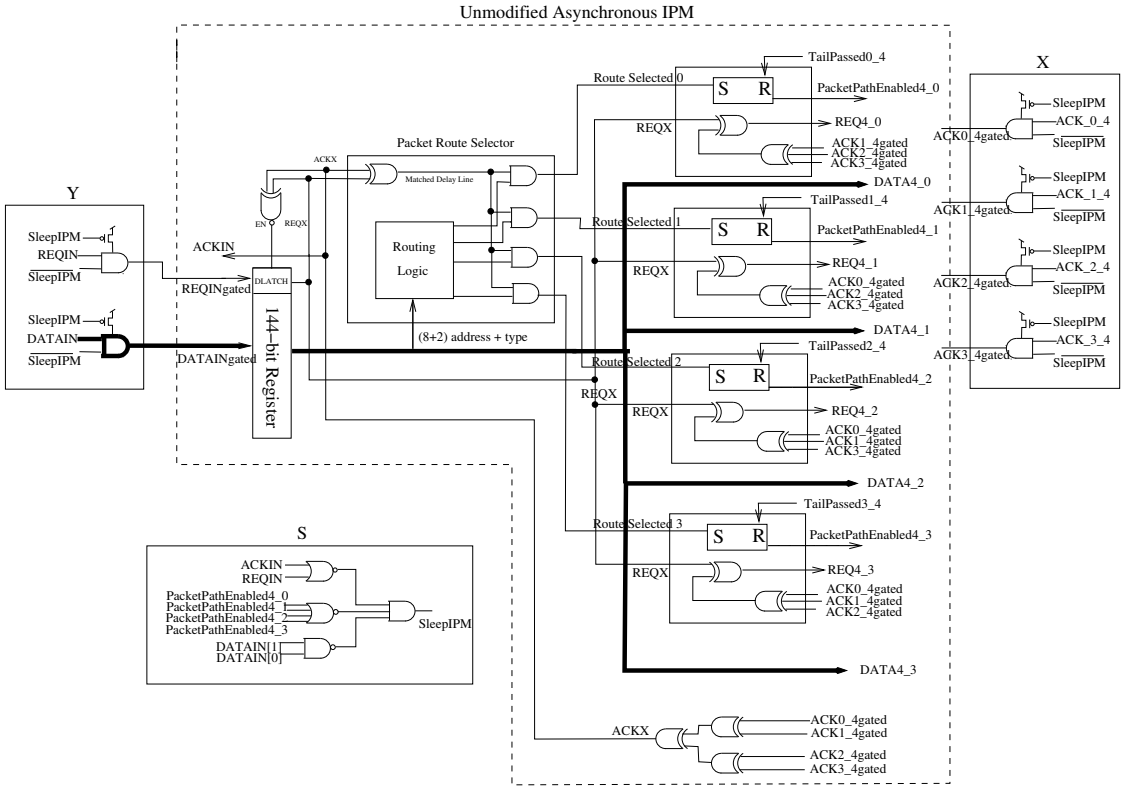


Fig. III.9. Input Port Module (IPM4)

The low-leakage HL version of the IPM circuit uses the circuits labeled X, Y and S in Figure III.9. As mentioned in Section III-D, each packet has an even number of flits, to maximize the leakage reduction opportunities.

Circuit S shows that when all *PacketPathEnabled4_i* signals are low (no flits being routed), *REQIN* and *ACKIN* are both zero (no packet being routed), and the 2 LSBs of *DATAIN* are not "1" each (no header flit detected), the IPM sleep signal is asserted. When a new packet arrives (header flit is driven on *DATAIN*), the IPM sleep signal is driven low. The power supplies are restored to their rail values during this wake-up phase. Once they have been restored, the *REQIN* signal is driven high, to begin the process of routing in an identical manner as in the paragraph above (for the unmodified router). While flits are

being routed, one of the *PacketPathEnabled4_i* signals is asserted, preventing the IPM from sleeping. This essentially means that the router cannot sleep between flits, but only between packets. When the last flit of a packet passes through OPM1, the *TailPassed1_4* signal is driven high, driving *PacketPathEnabled4_1* low. At this time, the sleep signal is driven high again.

Circuits X and Y show that when the IPM is in standby, all 144 *DATAIN* signals, as well as the *REQIN* and the *ACKi_4* signals are driven low, with the L-type AND2 gates. This is because during sleep, the input vector we select for the IPM is the vector in which all the inputs to the IPM are ”0”. We refer to circuits X and Y as *forcing circuits*.

In our design, bits 10 through 143 of *DATAIN* are actually forced to a ”1” value during sleep mode, using a NAND gate (instead of an AND gate as shown in Figure III.9). These bits again get inverted when they pass through the forcing circuit in the OPM. Since bits 0 through 9 of *DATAIN* are used for packet type detection and routing purposes, they are forced to a ”0” value during sleep mode, using an AND gate as shown in Figure III.9.

Based on the sleep vector inputs, all the gates in the IPM (in the dotted box) are appropriately changed to H or L gates, for leakage reduction.

III-D.2. Modified Output Port Module (OPM) for Reduced Leakage

The role of the OPM is to forward flits to the next link, if it is selected by the IPM, based on the destination address of the packet being routed. The schematic of OPM4 is shown in Figure III.10. Other OPMs are identical, with slightly altered signal indices.

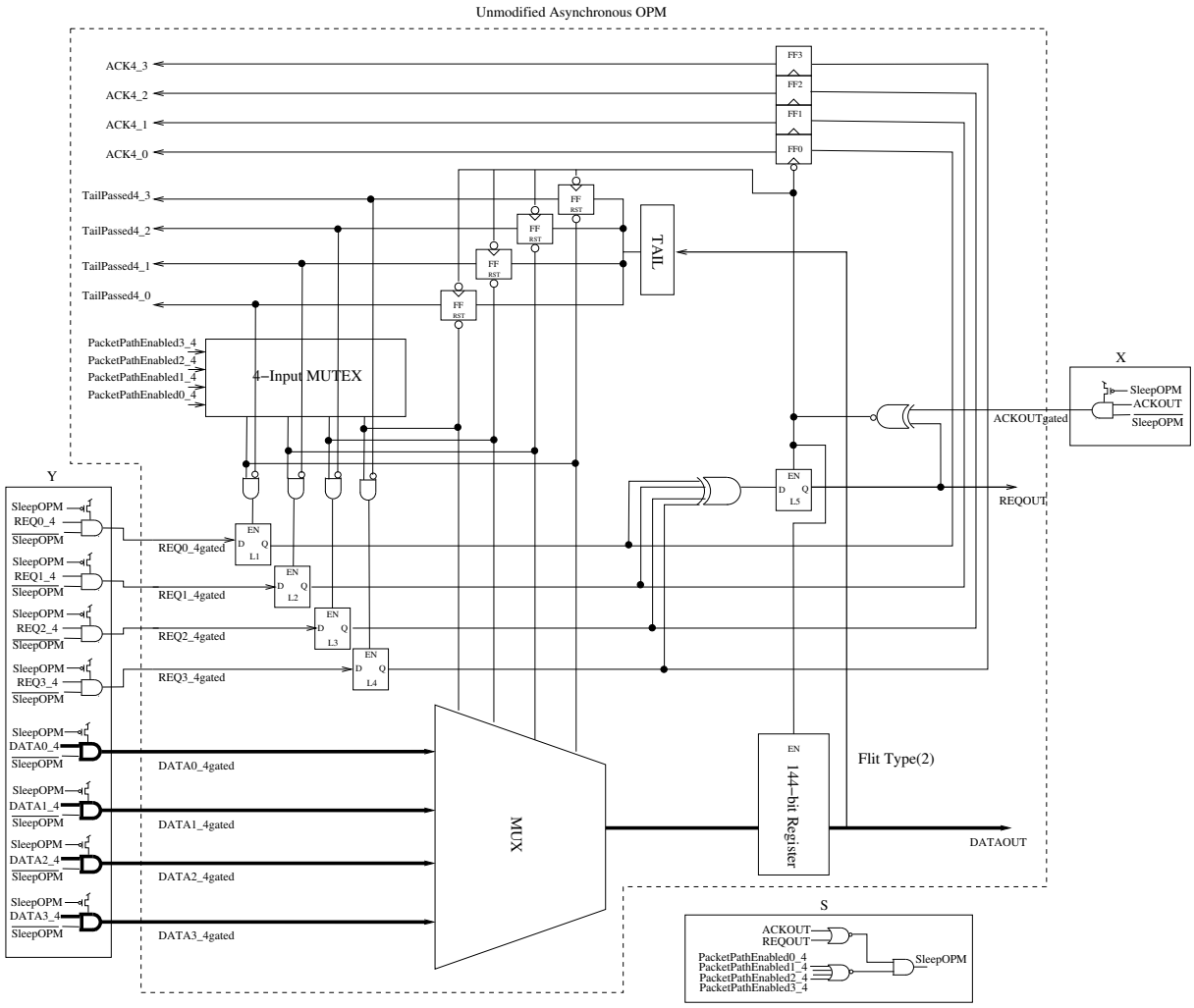


Fig. III.10. Output Port Module (OPM4)

The normal logic of the OPM is shown in the dotted region of Figure III.10. The modifications to the OPM to enable leakage reductions are shown in Figure III.10, in the solid boxes.

For the following discussion, we assume the destination address of the packet requires that it exit from OPM4, and the packet arrives from IPM1. As a consequence, *PacketPathEnable1_4* is high. Suppose that one or more other *PacketPathEnablei_4* are also high. In this case, the 4-input MUTEX block arbitrates which IPM gets to drive it's packet.

The 4-input MUTEX circuit has 4 *grant_i* signals which are shown at the bottom of it's circuit block in Figure III.10. The 4-input MUTEX block is illustrated in Figure III.11, and in turn consists of three 2-input MUTEX circuits (Figure III.12) and four Muller C-elements (Figure III.13). A 2-input MUTEX uses low-switchpoint inverters and a cross-coupled pair of NAND gates, ensuring that only one of the outputs g_1 or g_2 are high. Table III.1 is the Truth Table of the 2-input MUTEX circuit. The C-elements drive a 1(0) value if both inputs are 1(0), otherwise the output is unchanged. The logic of the C-elements is shown in Truth Table III.2. The 4-input MUTEX essentially ensures that a single *grant_i* signal is driven from among the IPMs requesting access to OPM4. It also enables fair arbitration between all incoming request signals.

r1	r2	g1g2
0	0	00
0	1	01
1	0	10
1	1	01 or 10

Table III.1. Truth Table of 2-input MUTEX

I1	I2	C
0	0	0
0	1	No Change
1	0	No Change
1	1	1

Table III.2. Truth Table of C-element

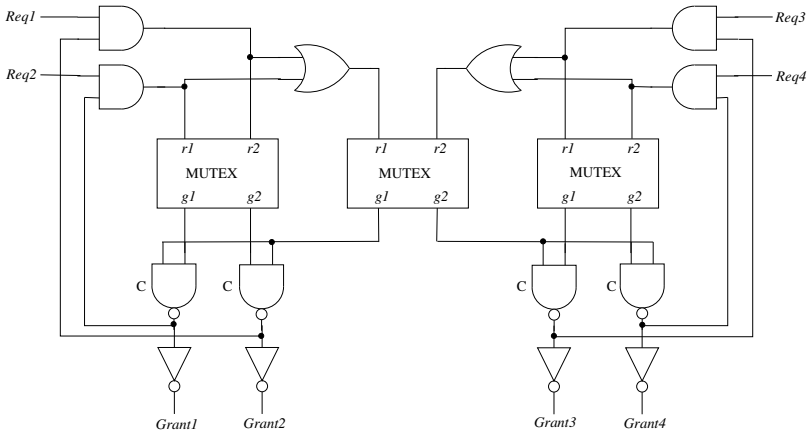


Fig. III.11. 4-input MUTEX Circuit

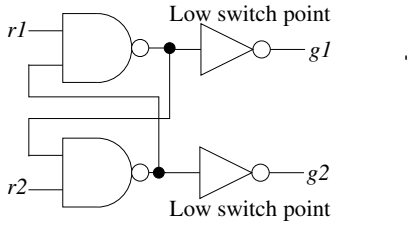


Fig. III.12. 2-input MUTEX Circuit

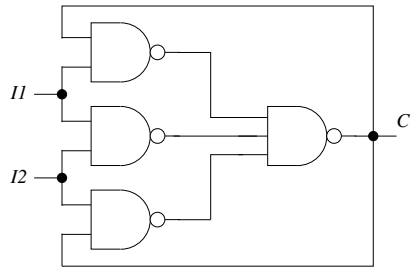


Fig. III.13. C-element Circuit

The selected IPM (assume it is IPM1 for our discussion) latches its *REQ1_4* signal, and if the *ACKOUT* of the OPM is different from it's *REQOUT*, the data (*DATA1_4*) is latched and driven out of the OPM as *DATAOUT*. Simultaneously, *REQOUT* changes, signaling the next link that data is available, while IPM1 is sent an acknowledge signal *ACK4_1* so it can send the next flit. Once the last flit of a packet is detected (by means of the *Tail-Passed4_1* signal going high, the *PacketPathEnabled4_1* is driven low (by the IPM circuit, see Figure III.9).

The low-leakage HL version of the OPM circuit uses the circuits labeled S, X and Y in Figure III.10. As mentioned in Section III-D, each packet has an even number of flits, to maximize the leakage reduction opportunities.

Circuit S shows that when all *PacketPathEnabled4_i* signals are low (no flits being routed), *REQOUT* and *ACKOUT* are both "0" (no packet being routed), then the OPM sleep signal is asserted. When a flit is awaiting being routed through the OPM (i.e. at least one *PacketPathEnabled4_i* signal is high), the OPM is woken up, and kept awake until all *PacketPathEnabled4_i* signals are low. Since *PacketPathEnabled4_i* stays asserted for the length of the packet, wormhole routing is achieved, without the router going into standby between flits of the same packet. Similarly, if *REQOUT* is not equal to *ACKOUT*, the OPM stays awake.

Assuming the OPM is asleep, the rising of one or more *PacketPathEnabled4_i* will wake up the OPM. The power supplies are restored to their rail values during this wake-up phase. While flits are being routed, one of the *PacketPathEnabled4_i* signals is asserted, preventing the IPM from sleeping. This essentially means that the router cannot sleep between flits, but only between packets. Finally, after the entire packet has been routed, the OPM may sleep again when REQOUT equals ACKOUT (transaction completed) and all *PacketPathEnabled4_i* signals are low.

Circuits X and Y show that when the OPM is in standby, all 144 *DATAi_4* signals (for $i = 1,2,3,4$), as well as the *REQi_4* and the *ACKOUT* signals are driven low, with L-type AND2 gates. This is because during sleep, the input vector we select for the OPM is the vector in which all the inputs to the OPM are "0". We refer to circuits X and Y as *forcing circuits*.

Just like in the IPM, bits 10 through 143 of *DATA_i_4* are actually forced to a "1" value during sleep mode, using a NAND gate (instead of an AND gate as shown in Figure III.10). Recall that these bits were inverted when they passed through the forcing circuit in the IPM. Since bits 0 through 9 of *DATA_i_4* are used for packet type detection and routing purposes, they are forced to a "0" value during sleep mode, using an AND gate as shown in Figure III.10.

Based on the sleep vector inputs, all the gates in the OPM (in the dotted box) are appropriately changed to H or L gates, for leakage reduction.

III-D.3. Discussion on HL versus MTCMOS, Header-only and Footer-only Approaches

There are two key benefits to using HL based leakage control versus the MTCMOS, Header-only or Footer-only approaches.

III-D.3.a. Sleep Inverter Chain Optimization

Consider the inverter chain required to drive the sleep signal for a design, as shown in Figure III.14 a). MP and MN are the high-threshold PMOS and NMOS transistors respectively. In general, for MTCMOS, Header-only and Footer-only, it is impossible to connect the 4 inverters I1 through 14 (which buffer the sleep signal) to MP and MN power gating transistors. This is because once the circuit is in sleep mode, the 4 inverters would be non-functioning, and the circuit would not be able to exit sleep mode again. As a consequence, these inverters would need to be always-awake, and hence would leak significantly, especially since they are typically large since they drive a large load.

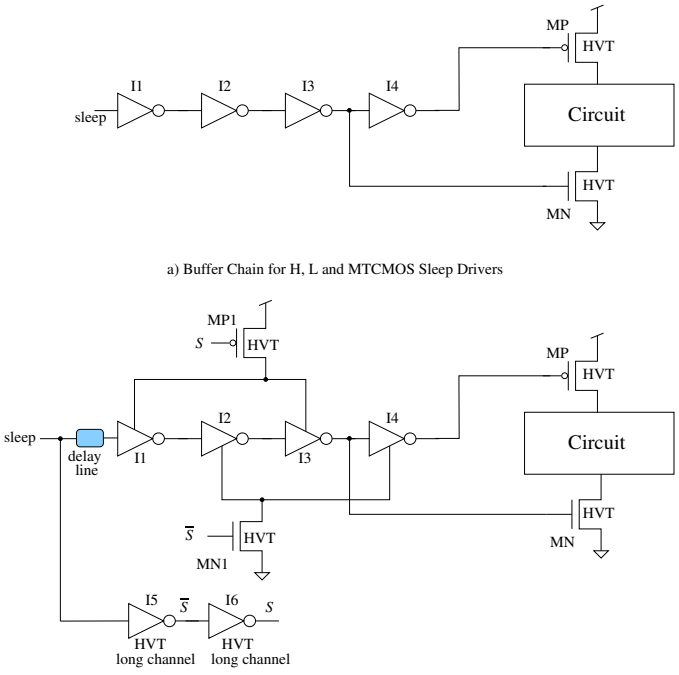


Fig. III.14. Sleep Driver Chain Optimization in our Approach

Now, with the HL approach, we can power gate these 4 devices as well, thus reducing leakage power further. Consider the circuit shown in Figure III.14 b). In this circuit,

I1 and I3 are connected to a high-threshold voltage power gating PMOS device MP1. I2 and I4 are connected to a high-threshold voltage power gating NMOS transistor MN1. Two additional always-on inverters (implemented with long-channel, high-threshold voltage devices to reduce leakage power) I5 and I6 are connected to the sleep signal. These inverters drive the gate of MN1 and MP1 respectively. The sleep signal is delayed (using a pair of high-threshold voltage, long channel inverters) before it drives the inverter chain I1 through I4.

Now consider the case that the circuit is awake, and going into sleep mode (i.e. the signal sleep is rising). MP1 and MN1 immediately go to sleep, since they are driven (undelayed) by S and $\overline{S}$, the outputs of I6 and I5 respectively. However, after a delay D (the delay of the delay line), the output of I1 still falls since it is connected to the ungated ground signal. Similarly, the output of I2 rises, and the outputs of I3 falls as well. Finally, the output of I4 rises. Now the power gating devices of the circuit (MP and MN) turn off, and the circuit enters sleep mode.

Consider the case that the circuit is in sleep mode, and is exiting sleep (i.e. the signal sleep falls). In this case, I5 and I6 immediately cause MN1 and MP1 to exit sleep. After a delay D, the sleep signal now drives I1 through I4 in turn (since they are now awake), causing MP and MN to exit sleep as desired.

A similar circuit cannot be devised for the MTCMOS, Header-only or Footer-only approaches. When the sleep signal rises, the inverter chain would immediately enter sleep mode, making it impossible for MN and MP to enter the sleep state. For these alternate leakage approaches, therefore, the inverter chain driving the sleep signal needs to be always awake, increasing leakage power significantly (since the inverter chain typically contains large inverters on account of the large size of MP and MN).

Without the use of the circuit of Figure III.14 b), our router achieved a $\sim 6\times$ leakage improvement over the unmodified router. With the use of the circuit of Figure III.14 b), this

number improved to 8.13×.

III-D.3.b. Fine-grained Per- IPM and OPM Sleep Ability

In our HL approach, the sleep functionality of the router is achieved in a per-IPM and per-OPM fashion, as described in Section III-D.1 and Section III-D.2. However, in the MTCMOS, Header-only or Footer-only approaches, the sleep functionality is only possible to implement on a coarser, per-router basis. This is because the sleep logic for the IPM and OPM (the circuit marked ”S” in Figure III.9 and Figure III.10) requires internal router signals (such as PacketPathEnabled4_i, which, in the MTCMOS, Header-only or Footer-only approaches is a floating signal). The ability to sleep on a per-IPM or per-OPM basis would avail a much greater leakage reduction under normal operation of the CMP.

CHAPTER IV

EXPERIMENTAL RESULTS

We implemented the asynchronous NoC router of [2] (henceforth *original router*), along with our modified HL-based low-leakage asynchronous NoC router (henceforth *HL router*) in HSPICE [29], in a 45nm PTM [10] high-performance process. We also conducted our experiments using the low-power 45nm PTM process cards, and obtained much higher leakage improvements, but with router delays (for the unmodified as well as the modified designs) that were about $3\times$ larger. Since a NoC router needs to be able to route packets at high speed, we focus on the results from the simulations with the high-performance model cards.

The first step was to design both the original router in Verilog. To obtain the HL router, the IPMs and OPMs of the original router were driven with a sleep vector in which all the inputs had the ”0” value. The router was simulated logically, and all cells with a ”0” output were replaced by L cells, and those with ”1” outputs were replaced by H cells.

After this, we thoroughly simulated both the original router and the HL router to verify logical correctness. Both routers were placed in a 4×4 NOC, and were tested at the logical level to ensure correct routing functionality. In the case of the HL router, correct assertion of the sleep signal was verified as well. A flowchart illustrating these steps is presented in Figure IV.1.

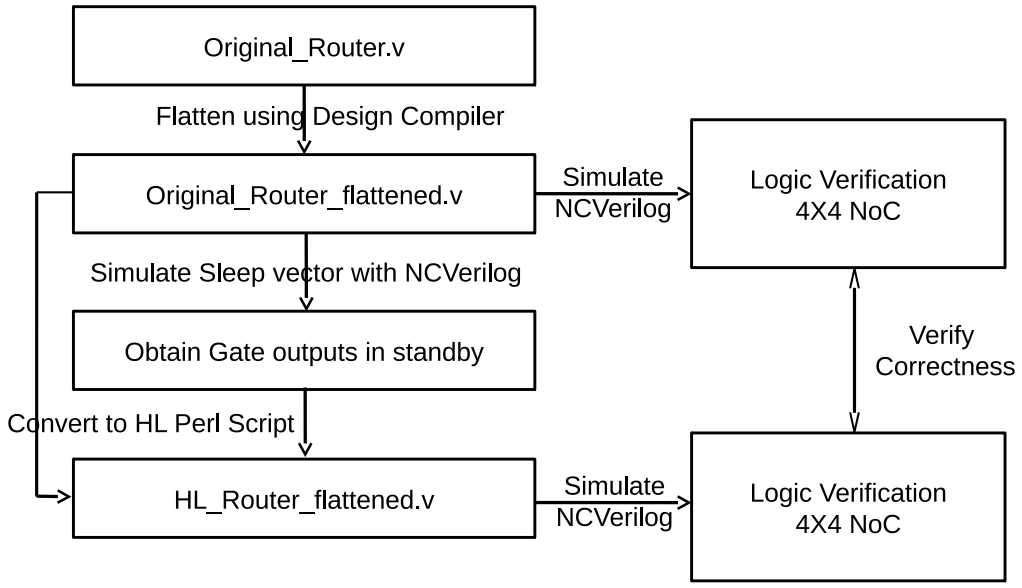


Fig. IV.1. Experiments Flowchart Part1

The Verilog netlists of both the original router and the HL router were next converted to HSPICE netlists using the V2S [30] tool.

Next, we sized the PMOS header and NMOS footer devices for both the IPMs and OPMs of the HL router. While the router was routing packets in HSPICE, we sized the headers and footers so that there was at most a 100mV droop in the supply signal, and a 100mV bounce in the ground signal of any OPM or IPM. The data flits being driven through the router were designed for maximal stress on the power and ground networks. The threshold voltage of the headers and footers were 200 mV above the nominal threshold voltage of PMOS and NMOS devices in the design.

Next, we simulated the original and HL routers to measure their delays, dynamic power, static power, and wake-up time (for the HL router). These steps are summarized in Figure IV.2. The results of these HSPICE simulations are presented next.

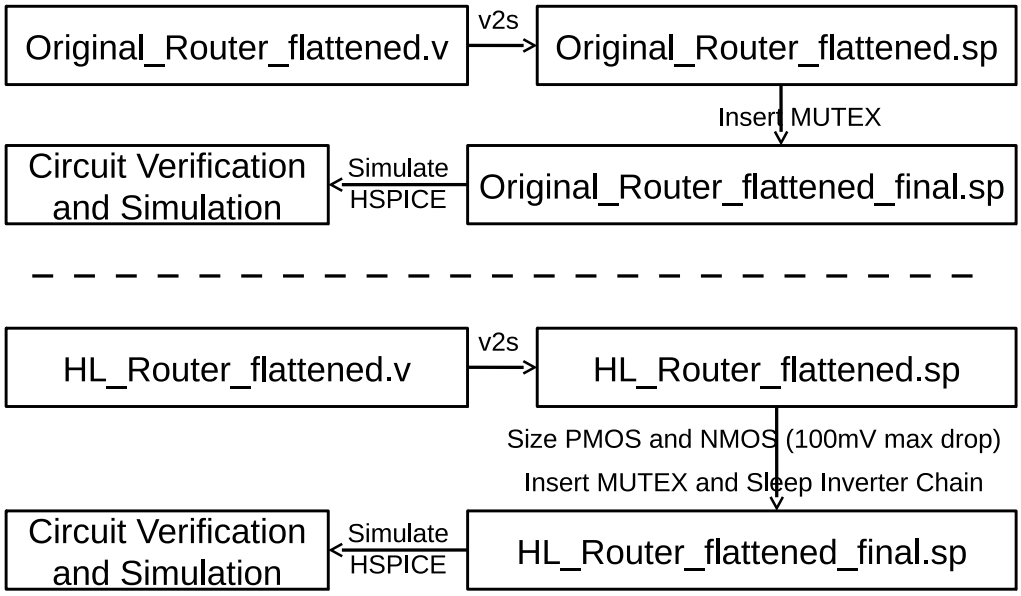


Fig. IV.2. Experiments Flowchart Part2

Note that the dynamic power numbers that are reported are the maximum value of dynamic power, when packets are being routed through every IPM and OPM (in other words, 5 packets are being routed through the asynchronous router), with maximal switching of the bits between flits being routed.

The leakage power numbers that are reported are for the condition when all IPMs and OPMs are in the sleep state. Note that for our HL router, the sleep functionality is imple-

mented on a per-IPM and per-OPM basis, so our router can benefit from a ”partial” sleep condition in which some IPMs and OPMs are active, while others are inactive (and can thus enter sleep mode). In the case of the MTCMOS, Header-only and Footer-only approaches to leakage control, the sleep functionality can only be implemented on a per-router basis, dramatically reducing the ability to avail of opportunistic leakage power reductions when only some IPMs and OPMs are active.

Design	Delay	Dynamic Power		
	Total	IPM	OPM	Total
Original Router	944.5 ps	6.55 mW	6.73 mW	13.28 mW
HL Router	2.73×	0.44×	0.63×	0.54×

Table IV.1. Delay and Dynamic Power Comparison for First Flit

Design	Delay	Dynamic Power		
	Total	IPM	OPM	Total
Original Router	543.6 ps	2.04 mW	1.71 mW	3.75 mW
HL Router	1.13×	0.85×	1.13×	0.98×

Table IV.2. Delay and Dynamic Power Comparison for Subsequent Flits

Table IV.1 reports the delay and dynamic power for the first flit of any packet. Table IV.2 reports the delay and dynamic power for the subsequent flits of any packet. Note that for all tables, the actual numerical values are presented for the original router, and relative values are presented for our router. We note that the delay for the first flit is about 2.73× higher for our approach. This is because the time for the IPM to wake up is included in this delay. In general, the delay for the first flit is higher than for subsequent flits, since the first flit requires address decoding and path setup. Importantly, the delay overhead for

our approach for subsequent flits is only 13%. The dynamic power consumption for the first flit for our router is about 0.54× that of the original router, while the dynamic power consumption for subsequent flits for our router is about 98% that of the original router.

Design	Leakage Power			Wakeup
	IPM	OPM	Total	Time
Original Router	98 μW	180.9 μW	278.9 μW	NA
HL Router	1/8.7 ×	1/7.85 ×	1/8.13 ×	873 ps

Table IV.3. Leakage Power and Wakeup time

Table IV.3 reports the leakage numbers and wake-up times for the asynchronous routers. Our router achieves a 8.13× improvement in leakage compared to the original router. The wake-up time of our router was about 870 ps. This number can be improved further at the cost of some leakage improvement. However, this number is less than the cycle time of a typical 1 GHz CMP, and is quite acceptable in practice.

Design	Area		
	IPM	OPM	Total
Original Router	163.94 μ^2	361.72 μ^2	525.66 μ^2
HL Router	1.23×	1.23×	1.23×

Table IV.4. Router Area

Table IV.4 reports the active area numbers of the IPMs and OPMs. We note that the 8.13× leakage improvement of our approach comes at a nominal 23% area cost.

Component	Area (μ^2)	Percentage
Unmodified Router	525.66	81.40
Forcing Circuits	73.82	12.13
Sleep Generation Circuit	2.88	0.45
Sleep Inverter Chains	17.69	2.74
Header and Footer Devices	21.26	3.29
Total	645.81	100

Table IV.5. Area Breakdown for HL Router

Table IV.5 reports the area breakdown of the various components of our router. Note that the gates in the forcing circuits contribute the most to the area overhead of our approach.

CHAPTER V

CONCLUSION

In this work, we take a state-of-the-art asynchronous router for a Network-on-Chip (NoC), and modify it for leakage improvement. We use a powerful leakage control technique, which allows internal circuit nodes to stay at rail values during sleep mode. We present a novel leakage reduction approach for the inverter chains that drive the sleep signal, further improving the leakage reductions obtained. This technique cannot be used in traditional leakage control approaches. We implemented and verified (both at the logical and circuit levels) the unmodified and the modified routers, and show that the modified router obtains a 8.13$\times$ reduction in leakage in sleep mode, with a 13% delay penalty and a low wake-up delay.

REFERENCES

[1] Michael Taylor, Michael Bedford Taylor, Walter Lee, Saman Amarasinghe, and Anant Agarwal, "Scalar operand networks: On-chip interconnect for ilp in partitioned architectures," in *International Symposium on High Performance Computer Architecture*, 2002, pp. 341–353.

[2] A Ghiribaldi, D Bertozzi, and S Nowick, "A transition-signaling bundled data NoC switch architecture for cost-effective GALS multicore systems," in *Proceedings, Design Automation and Test in Europe (DATE) Conference*, Grenoble, France, March 2013, pp. 332–337.

[3] P Teehan, M Greenstreet, and G Lemieux, "A survey and taxonomy of gals design styles," *IEEE Design and Test of Computers*, vol. 24, no. 5, pp. 418–428, Sept. 2007.

[4] Chenming Hu and Ali Niknejad, "Bsim4 homepage," http://www-device.eecs.berkeley.edu/bsim/?page=BSIM4, May 2013.

[5] Wolfgang Arden, Patrick Cogez, and Mart Graef, "The International Technology Roadmap for Semiconductors," `http://public.itrs.net/`, 2003.

[6] M Mui, K Banerjee, and A Mehrotra, "Power supply optimization in sub-130 nm leakage dominant technologies," in *Proceedings of the International Symposium on Quality Electronic Design (ISQED)*, Washington, DC, USA, 2004, pp. 409–414.

[7] J Rabaey, *Digital Integrated Circuits: A Design Perspective*, Prentice Hall Electronics and VLSI Series. Prentice Hall, 1996.

[8] N Jayakumar and S Khatri, "A predictably low leakage ASIC design style," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 15, no. 3, pp. 276–285, Mar 2007.

[9] S. Stergiou, F. Angiolini, S. Carta, L. Raffo, D. Bertozzi, and G. De Micheli, "Xpipes lite: A synthesis oriented design library for networks on chips," in *Proceedings, Design Automation and Test in Europe (DATE) Conference*, Munich, Germany, March 2005, pp. 1188–1193.

[10] Arizon State University, "Predictive Technology Model," `http://ptm.asu.edu`, May, 2013.

[11] Y. Thonnart, P. Vivet, and F. Clermidy, "A fully-asynchronous low-power framework for GALS NoC integration," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, 2010, pp. 33–38.

[12] T. Bjerregaard and J Sparso, "A router architecture for connection-oriented service guarantees in the MANGO clockless network-on-chip," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE) - Volume 2*, Munich, Germany, 2005, pp. 1226–1231.

[13] A. Sheibanyrad, A. Greiner, and I. Panades, "Multisynchronous and fully asynchronous NoCs for GALS architectures," *IEEE Design & Test of Computers*, vol. 25, no. 6, pp. 572–580, 2008.

[14] M. Singh and S. Nowick, "MOUSETRAP: High-speed transition-signaling asynchronous pipelines," *IEEE Trans. Very Large Scale Integrated Systems*, vol. 15, no. 6, pp. 684–698, June 2007.

[15] W.J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Design Automation Conference, 2001. Proceedings*, 2001, pp. 684 – 689.

[16] Jose Duato, Sudhakar Yalamanchili, and Ni Lionel, *Interconnection Networks: An Engineering Approach*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA,

2002.

[17] F. Karim, A. Nguyen, and S. Dey, "An interconnect architecture for networking systems on chips," *Micro, IEEE*, vol. 22, no. 5, pp. 36 – 45, Sep/Oct 2002.

[18] J. Kim, J. Balfour, and W.J. Dally, "Flattened butterfly topology for on-chip networks," *Computer Architecture Letters*, vol. 6, no. 2, pp. 37 –40, Feb. 2007.

[19] Charles E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," *IEEE Transactions on Computers*, vol. 34, pp. 892–901, October 1985.

[20] H. Samuelsson and S. Kumar, "Ring Road NoC architecture," in *Norchip*, 2004, pp. 16–19.

[21] F Assaderaghi, D Sinitsky, S A Parke, J Bokor, P K Ko, and C Hu, "Dynamic threshold-voltage MOSFET (DTMOS) for ultra-low voltage VLSI," *IEEE Transactions on Electron Devices*, vol. 44, no. 3, pp. 414–422, Mar 1997.

[22] T Inukai, T Hiramoto, and T Sakurai, "Variable threshold voltage cmos (VTCMOS) in series connected circuits," in *International Symposium on Low Power Electronics and Design*, 2001, pp. 201–206.

[23] Im Hyunsik, T Inukai, H Gomyo, T Hiramoto, and T Sakurai, "VTCMOS characteristics and its optimum conditions predicted by a compact analytical model," in *International Symposium on Low Power Electronics and Design*, 2001, pp. 123–128.

[24] J T Kao and A P Chandrakasan, "Dual-threshold voltage techniques for low-power digital circuits," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 7, pp. 1009–1018, Jul 2000.

[25] Z Chen, M Johnson, L Wei, and W Roy, "Estimation of standby leakage power in CMOS circuit considering accurate modeling of transistor stacks," in *International Symposium on Low Power Electronics and Design*, 1998, pp. 239–244.

[26] K. Gulati, N Jayakumar, and S Khatri, "A probabilistic method to determine the minimum leakage vector for combinational designs," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, Kos, Greece, May 2006, pp. 2241–2244.

[27] M. Imai, K. Takada, and T. Nanya, "Fine-grain leakage power reduction method for m-out-of-n encoded circuits using multi-threshold-voltage transistors," *IEEE 18th International Symposium on Asynchronous Circuits and Systems*, vol. 0, pp. 209–216, 2009.

[28] C Ortega, J Tse, and R Manohar, "Static power reduction techniques for asynchronous circuits," in *IEEE Symposium on Asynchronous Circuits and Systems (ASYNC)*, May 2010, pp. 52–61.

[29] Synopsys Inc, "HSPICE Quick Reference," 2013.

[30] Synopsys Inc, "V2s tool," www.synopsys.com, 2013.